International Journal of Computer Science Engineering
and Information Technology Research (IJCSEITR)
ISSN(P): 2249-6831; ISSN(E): 2249-7943
Vol. 7, Issue 3, Jun 2017, 1-10

TRANS STELLAR
•Journal Publications •Research Consultancy

EVOLUTION OF NEW EFFICIENT LOAD BALANCER ALGORITHM FOR LARGE DATA SETS

NUKA SANTOSH KUMAR[1], MAYANK KUMAR[2] & NISHI YADAV[3]

[1,2]B.Tech, Department of CSE, Institute of Technology, Guru Ghasidas University, Bilaspur, India

[3]Assistant Professor, Department of CSE, Institute of Technology, Guru Ghasidas University, Bilaspur, India

ABSTRACT

Cloud computing is a vital part of this new era IT world or we can say that it is a technology of new age which are used to connect data and application from anywhere around the planet through the internet. Anything and everything from servers to mobile phones can be connected to the cloud. It has also yielded up some new companies which are providing consumer a large range of services. Due to this the upcoming companies can now better concentrate on its major purpose and not have to worry about installing their own servers and about its maintenance. They can be very easily outsourcing it to these cloud suppliers saving a lot of capital and energy investment by them. Now, since the devices are increasing very fast a major matter of concern that needs an immediate attention has emerged and this is load balancing of rapid increasing load on the servers of cloud. All the resources that are connected to the cloud need to optimize its functionality so that the cloud providers can get proper ROI that is Return on Investment and resulting the cloud users worthwhile experience.

This paper introduces a better load balancing model based on cloud partitioning concept for the public cloud with a switch mechanism so that there is a provision to choose different strategies for any type of different situations. This paper focuses on applying game theory to the load balancing strategy for improving the efficiency in the public cloud environment for large data sets.

KEYWORDS: Cloud Computing, VM-Virtual Machine & Balancer

Received: Mar 21, 2017; **Accepted:** Apr 05, 2017; **Published:** Apr 13, 2017; **Paper Id.:** IJCSEITRJUN20171

Original Article

INTRODUCTION

A virtual machine is a virtual form of computer hardware within software. Virtual machine is a software implementation that executes programs as if they are actual physical machines. Installing new applications and storing the data on a local drives are scenario of past, now users don't need to install these applications or store data, all these stuffs are taken care by cloud providers. Cloud will change the IT world it is said in Gartner's report [1]. Cloud computing is scalable and efficient but for maintaining its stability there are so many jobs which are very complex in themselves and that is why load balancing is receiving much attention by researchers.

Load balancing depends whether the system is dynamic [2] or static. Dynamic schemes are more complex and use system information whereas static schemes are bit simpler and system information is not used here. Dynamic scheme get changed whenever the status of the system gets changed. In our paper we have used dynamic schemes for flexibility. In our model there is a main controller and balancer which gather and analyze all the information. We aimed at the public cloud in this load balancing model because public cloud has enormous nodes with distributed computing resources present in many different locations. In this model public cloud is being di-

vided into several cloud partitions and when the load and environment is very large and complex, these partitions simplify the load balancing. Due to its main controller the cloud can choose suitable partition for the jobs which are arriving whereas the balancer for each cloud partition chooses the best load balancing strategy.

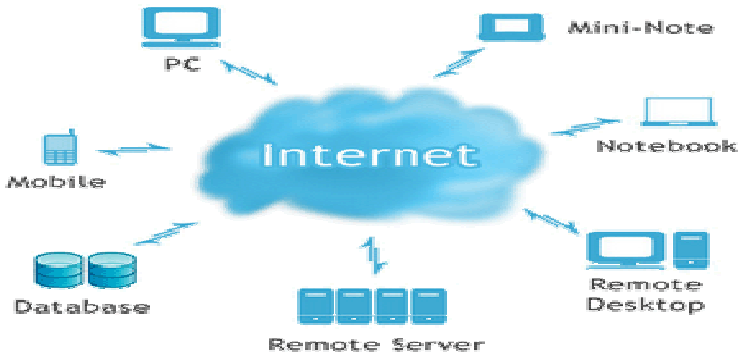

Figure 1: Representation of Cloud Computing [16]

RELATED WORKS

There have been many studies of load balancing for the cloud environment. Load balancing in cloud computing was discussed in a white paper by Adler [3] who introduced the tools techniques commonly used for load balancing in cloud. However load balancing is still a new problem in cloud computing that needs new architectures to adapt the changes. Chaczko et al [4] described the role that load balancing plays in maintaining stability and improving performance

There are so many algorithms for cloud computing like round robin, ant colony ESCE (equally spread current execution) algorithm etc. Nishant et al [5] used the ant colony algorithm for optimization in nodes. Randles et al [6] by checking the performance time and cost gave a compared analysis of some algorithms in cloud computing. Conclusion of their study was that round robin algorithm is fairly simple and better than ESCE algorithm and throtted algorithm, some of the conventional load balancing methods are similar to the allocation method in the operating system for example Round robin and FCFS rules. We have used round robin here for our work due to its simplicity and better running mechanism

- **Round Robin [7]:** In round robin, number of requests is done on the data-centers on a rotation basis to the list of Virtual machines. In this algorithm virtual machine is selected randomly and then the request is assigned to it, after those successive requests are assigned to VMs in a circular order, that is, once the VM is assigned the request then VM id is moved to the end of the list. This is how the Balancer of the round robin works.
- **Throttled Load Balancing[8]:** This algorithm maintains the status of every Virtual machine and send required virtual machine whenever and wherever needed that is it maintain records whether the Virtual machine is idle/busy and whenever a request is made for Virtual machine, ID of the idle VM is send by Throttled Load Balancing to the data center controller and after that, data center controller allocate that idle virtual machine and if virtual machine is busy it waits till the status changed to idle.
- **Least-Connection [9]:** The Least connection scheduling algorithm directs network connection to the servers with the least number of established connections. This is one of the dynamic scheduling algorithms as it needs to count live connections for each server dynamically.

- **Min-Min Algorithm [10]:** It starts with a set of all unassigned tasks. At first, minimum completion time is found for all tasks. Then the minimum value is selected among these minimum times which are also having minimum time among all the tasks on any resources. Then on the basis that minimum time, the task is scheduled on the corresponding machine. After that the execution time for all other tasks is updated on that machine by adding the execution time of the assigned task to the execution times of other tasks on that machine and assigned task is removed from the list of the tasks that are to be assigned to the machines. This procedure is repeated until all the tasks are assigned. But this approach has a major drawback that it can lead to starvation.

Round Robin Algorithm: Round-robin load balancing is one of the simplest methods for distributing requests of the client across a group of servers. It is one of the simplest scheduling techniques that utilize the principle of time slices. Here the time is divided into multiple slices and each node is given a particular time slice or time interval that is it utilizes the principle of time scheduling. The algorithm is as follows:

Step 1

Round robin Virtual Machine load balancer maintains an index of VM (virtual machines) and state of the Virtual Machines (busy/available). At the beginning all Virtual Machines have zero allocation.

Step 2

- Request/cloudlets are received by the data center controllers.
- After this the arrival time and burst time of the user requests is stored.
- On the basis of their states known from the virtual machine queue the request is allocated to virtual machines
- The round robin virtual machine load balancer will allocate the time quantum for user request execution.

Step 3

- The round robin virtual machine load balancer will calculate the turn- around time of each process
- It also calculates the response time and average waiting time of user requests.
- Scheduling order is then decided.

Step 4:

After the execution of cloudlets, the virtual machines are re-allocated by the round robin virtual machine load balancer.

Step 5

New/pending/waiting requests in queue is then checked by the data center controller

Step 6

Execution will again continue from step2.

PROPOSED ALGORITHM

Choosing right partition is the first step when a job arrives at the public cloud. There are three types in which par-

tition status can be divided.

- idle
- Normal
- Overload

When job i arrives at the system, the main controller queries about the cloud partition where job is located. The job is handled only if this location's status is idle or normal and, if not, another cloud partition is found that is not overloaded. Pseudo code is shown below:

Pseudo Code:

Best Partition Searching

start

while load do

Best Partitions each (load);

if State of partition == idle || State of partition == normal then

Send load to Partition;

else

search for next Partition;

end if

end while

end

Loads Assignment to the Nodes of the Cloud by Using Partition Status Calculation

Cloud partition balancer gathers load information from every node to evaluate the status of the cloud partition. This evaluation of each node's load status is very important. But prior to this evaluation of load degree of each node is defined. Load degree of node is related to various dynamic and static parameters. The static parameters include the CPU Processing speeds, number of CPU's, memory size, etc. Dynamic parameters are the network bandwidth, memory utilization ratio, CPU utilization ratio etc.

The load Degree is computed from these parameters as below:

Step 1

Define a load parameter set: P= {P1, P2…P_m}

With each P_i (1< i <m, P_i belongs to [0 and 1])

Notations used:

m:total no. of parameters

P: different sets of static and dynamic parameters

Step2

Compute the load degree as: load degree (N) = $a_1P_1+a_2P_2+...+a_nP_n$

n = total no. of loads

a_i = weights that may differ for different kind of jobs.

N = The current node.

Step3

In this step evaluation benchmark is defined. The average cloud partition degree is calculated from the node load degree statistics as:

n* load degree $_{avg}$ =load degree(N1) + load degree(N2)+…+load degree(N_n).

Based on the load degree $_{avg}$ the benchmark load degree $_{high}$ is set for different situations

Step 4

Three node load status levels are then defined as:

>Idle: when

Load degree (N) is equal to zero

The status is charged to idle as there is no jobs to be processed by the node.

>Normal: when

Load degree of the node is greater than zero and less than or equal to load degree $_{high}$ that is

0< load degree (N) <=load degree $_{high}$ the node is charged to normal and it can process other jobs

>Overloaded: when

Load degree of the current node is greater than load degree $_{high}$ that is:

Load degree $_{high}$ <=load degree (N)

Node will not be available to receive jobs or loads until it returns to its normal state. The cloud partition balancers create the load status tables and the load degree results are inputted into it and get refreshed after every fixed time period T

The partition status is then calculated by the balancers using the table inputs. Every partition status has a different load balancing solution. The balancer assign job to the nodes based on its current load strategy. Strategy is changed by the balancer as the cloud partition status changes

Load Balancing Strategy for the Idle Status

When the cloud partition is idle, it implies that many computing resources are available and relatively few jobs are arriving. In this situation, this cloud partition has the ability to process jobs as quickly as possible so that simple load balancing method can be used. There are many simple load balance algorithm methods such as the Random algorithm, the

Dynamic Round Robin [12], and the Weight Round Robin. Among these one of the simplest algorithm is round robin and round robin algorithm has been used here for simplicity it passes each new request to the next server in the queue. The algorithm does not keep the status record of each connection so it has no status information. Every node has an equal opportunity to be chosen in the round robin algorithm. However, the configuration and the performance of each node will not be the same in a public cloud; thus, this method may overload some nodes. Thus, an improved Round Robin algorithm is used, which called "Round Robin using partition based on the evaluation of load degree". The algorithm is still very simple. In this new algorithm Before the Round Robin step, on the basis of the load degree the nodes in the load balancing table are ordered from the lowest to the highest. a circular queue is build by the system and system walks through the queue again and again. Low load degree nodes will then assigned new jobs. When the load status table will get refreshed the order of the node will be changed

When the Cloud Partition is Normal

Jobs arrive much faster than in the idle state and is far more complex situation, so a different strategy is used for the load balancing. Each user wants his jobs to be done in the shortest time, so the public cloud needs a method that can complete jobs of all users with reasonable response time. So a static load balancing strategy based on game theory for distributed systems was proposed by Penmatsa and Chronopoulos [13]. And this work provides us a new review of the load balance problem in the cloud environment. As an implementation of distributed system, the load balancing in the cloud computing environment can be viewed as a game. In this Game theory it has cooperative games and non-cooperative games. In cooperative games, concept of binding agreement is there in which the decision makers decides by comparing notes with each others. In non-cooperative games, each decision maker makes decisions only for his own benefit. In this type of game theory each decision maker makes the optimized decision. In this type of game theory equilibrium is achieved when each player in the game has chosen a strategy and no player can be benefited by changing his or her strategy while the other player's strategies remain unchanged. Previous studies have shown that the load balancing strategy for a cloud partition in the normal load status can be viewed as a non cooperative game, as described here. The players in the game are the nodes and the jobs. Suppose there are n nodes in the current cloud partition with N jobs arriving, then we have to define the following parameters:

a_i: Processing ability of each node, i=1; _ _ _ ; n.

t_j: Time spending of each job.

$T=t_1+t_2+\ldots+t_{N,}$ time spent by the entire cloud

Partition, $t< a_1+a_2+\ldots+a_n$

f_{ji}: Fraction of job j that assigned to node i

To find an appropriate value of f_{ji} is the most important step in this model. The current model uses the method of Grosu et al.[14] called "the perfect reply" to calculate f_{ji} of each node, with a greedy algorithm and then used to calculate f_{ji} for all nodes. This procedure minimize the response time of each job. The strategy then changes as the node's status change

SIMULATION

For simulation we have proposed the Round Robin Load Balancer Algorithm using partition of the data according to the state of the virtual machines. We have used Java8.0 as platform for implementing VM load balancing algorithm. We

have also used cloud sim 3.0 toolkit for simulation purpose. Result for overall response time of the cloud based on round robin vm load balancing algorithm is shown in the table below. For different number of virtual machines we have analysed min (ms), max (ms) and avg (ms) time and also the execution time is being compared here.

Below table shows the execution time for different no. of virtual machines:

Table 1.1: Comparison between Average, Minimum and Maximum Time Vs Virtual Machines

No. of VM's used	AVG (ms)	min(ms)	max(ms)
10	400.5	400.2	400.8
15	400.7	400.6	400.8
20	400.8	400.2	401.4
25	400.9	400.6	401.2

*vm's: virtual machines

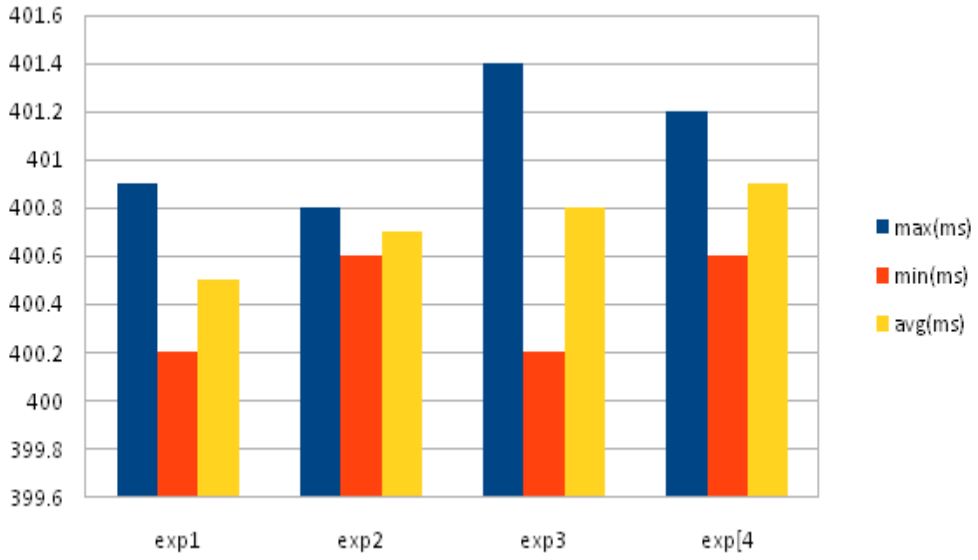

Figure 2

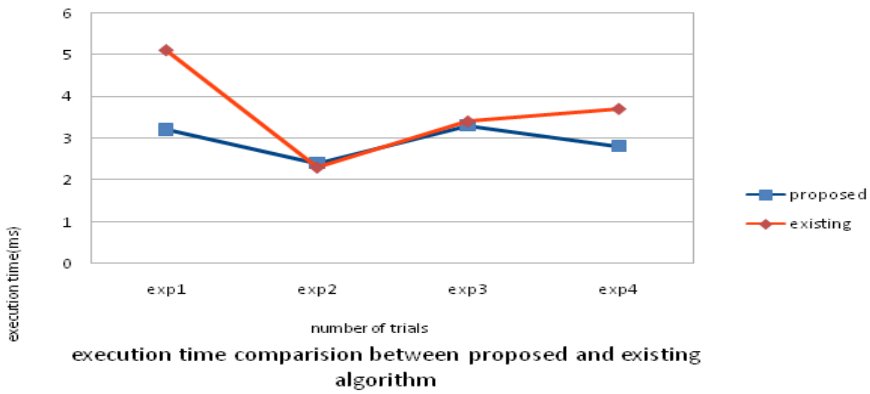

Figure 3

For comparing execution time we have done four trials by varying number of virtual machines. In this line chart we can clearly observe that average execution time of proposed algorithm is better than the existing algorithm due to utiliz-

ing the concept of job partitioning and distributing the jobs on the basis of present state of virtual machines

Round Robin Load balancing by partitioning method is consuming considerably lesser time as far as response and data processing time is concerned. Whereas the data processing time and response time increases when number of virtual machines increases. By the new server broker policy that is by partitioning the data according to the state of the virtual machine the problem of deadlock and server overflow decreases to a considerable extent.

CONCLUSIONS AND FUTURE WORK

We have done the analysis of existing scheduling algorithm and then we have proposed algorithm of Round Robin VM Load Balancing using partitioning of the data according to the state of virtual machines and for implementation we have used Java language for implementing VM scheduling algorithm in CloudSim3.0 toolkit. Assuming the application to be deployed in one data center having virtual machines. These experimental results shows that Round Robin VM Load Balancing method improves the performance by consuming less time for scheduling virtual machine for large data as com

pared to simple round robin algorithm ESCSE algorithm and throttled algorithm in cloud computing.

REFERENCES

1. *R. Hunter, The why of cloud, http://www.gartner.com/ Display Document? doc cd=226469&ref= g noreg, 2012*
2. *N. G. Shivaratri, P. Krueger, and M. Singhal, "Load distributing for locally distributed systems" vol. 25, no. 12, pp. 33-44, Dec. 1992*
3. *Mondal, B., K. Dasgupta, et al.. "Load balancing in cloud computing using stochastic hill climbing-a soft computing approach." Procedia Technology 4: 783-789,2012.*
4. *Dorigo, M., M. Birattari, et al.. "Ant colony optimization." Computational Intelligence Magazine, IEEE 1(4): 28-39,2006.*
5. *N. S. Raghava and Deepti Singh "Comparative Study on Load Balancing Techniques in Cloud Computing" OPEN JOURNAL OF MOBILE COMPUTING AND CLOUD COMPUTING Volume 1, Number 1, August 2014.*
6. *Dhurandher, S. K., M. S. Obaidat, et al., "A cluster-based load balancing algorithm in cloud computing." Communications (ICC), 2014 IEEE International Conference on, IEEE,2014.*
7. *Kaur, Sukhvir, and Supriya Kinger. "Review on Load Balancing Techniques in Cloud Computing Environment." IJSR, ISSN: 2319-7064,2007.*
8. *A. M. Alakeel, "A guide to dynamic load balancing in distributed computer systems," in: IJCSNS International Journal of Computer Science and Network Security,VOL.10 No.6., pp. 153–160,2010.*
9. *Karger, David R., and Matthias Ruhl. "Simple efficient load balancing algorithms for peer-to-peer systems." In Proceedings of the sixteenth annual ACM symposium on Parallelism in algorithms and architectures, pp. 36-43. ACM, 2004.*
10. *Xu, Zhiyong, and Laxmi Bhuyan. "Effective load balancing in p2p systems." In Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on, vol. 1, pp. 81-88. IEEE, 2006.*
11. *Karger, David, and Matthias Ruhl. "New algorithms for load balancing in peer-to-peer systems." (2003).*
12. *Qiao, Ying, and Gregor V. Bochmann. "Load balancing in peer-to-peer systems using a diffusive approach." Computing 94, no. 8-10: 649-678,2012.*
13. *Hsiao, H.-C., H.-Y. Chung, et al.. "Load rebalancing for distributed file systems in clouds." Parallel and Distributed Systems,*

IEEE Transactions on24(5): 951-962,2013.

14. *Karger, D. R. and M. Ruhl. "Simple efficient load balancing algorithms for peer-to-peer systems", Proceedings of the sixteenth annual ACM symposium on Parallelism in algorithms and architectures, ACM,2014.*
15. *Ajay Gulati1 and Ranjeev K. Chopra2, "Dynamic Round Robin for Load Balancing in a Cloud Computing", International Journal of Computer Science and Mobile Computing, volume 2, issue 6,, pg 274- 278,june 2013.*